170 praktických ukázek

CSS

MODERNÍ LAYOUT

Grid, flexbox a nové metody rozvržení webů v příkladech

Martin Michálek

CSS

MODERNÍ LAYOUT

Martin Michálek

ISBN: 978-80-88253-08-2

08

CO BYSTE JEŠTĚ MĚLI VĚDĚT?

V této kapitole opustíme refereční příručky a prohloubíme si znalosti souvislostí. Začneme u podpory v prohlížečích a přes přístupnost a rychlost se dostaneme k vývojářským nástrojům v prohlížečích, které nám mohou být velice nápomocny.

Internet Explorer (je už mrtvý?)

Microsoft Internet Explorer (MSIE) dělá vývojářům starosti. Používat moderní vlastnosti jako CSS grid je složitější a někdy i nemožné, pokud je na projektu nutné podporovat tento starý prohlížeč od Microsoftu.

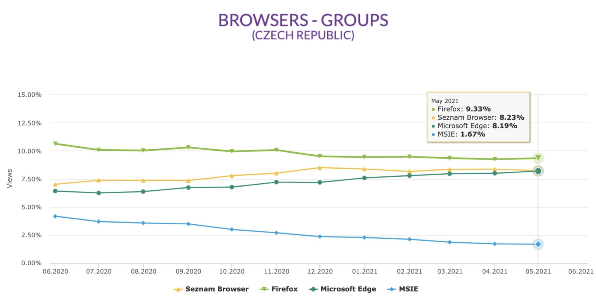
Pokud Internet Explorer stále ještě podporujete, myslím si, že byste od této praxe mohli začít upouštět. To je důvod, proč píšu tenhle text.

Velmi to záleží na cílové skupině konkrétního projektu, ale drtivá většina vývojářů si myslím ukončení podpory v roce 2022 může dovolit.

Toto je mé osobní rozloučení s prohlížečem, který do webdesignu přinesl mnoho dobrého. Ne, není to ironie. Jak brzy uvidíte, Explorer dříve stál na světlé straně Síly a možná vás překvapí, že to byl nejpokrokovější prohlížeč své doby.

Aktuální podíl MSIE mezi uživateli je v Česku mezi 1–2 %. A dál klesá

Podívejte se na graf. Je z něj myslím jasné, že i v České republice bude brzy možné přestat Internet Explorer podporovat.



Nejmenší z nejmenších. Podíl méně významných prohlížečů na trhu v ČR. Vývoj MSIE ukazuje světle modrá linka. Zdroj: rankings.gemius.com.

V Česku měl MSIE v polovině roku 2021 zastoupení kolem 1,5 % a začátkem roku 2022 už jen kolem 1 %. Tato čísla potvrzuje i jiný statistický web, StatCounter.

Daleko důležitější je ale trend vývoje. Zatímco začátkem roku 2019 používala dědeček prohlížeč ještě zhruba desetina uživatelů, zkraje roku 2020 už jej pro přístup na weby využívalo jen zhruba 5 %. Každý rok tedy jeho popularita klesne na polovinu a méně.

Ovšem bratrům Slovákům zde můžeme jen závidět. V zemi pod Tatrami je totiž zastoupení MSIE podle čísel Gemiusu už od roku 2021 nulové.

Výjimky potvrzují pravidlo. Sledujte statistiky a příjmy

Jak zjistit, jestli můžu Explorer přestat podporovat?

Podíl MSIE na používání se liší web od webu, takže například na svém blogu, Vzhůru dolů, nevidím v únoru 2022 prakticky žádné návštěvy od uživatelů s tímto prohlížečem. Na webech klientů ale Google Analytics ukazují čísla vyšší.

Podíl prohlížečů zjistíte například právě v Google Analytics (Publikum > Technologie > Prohlížeč).

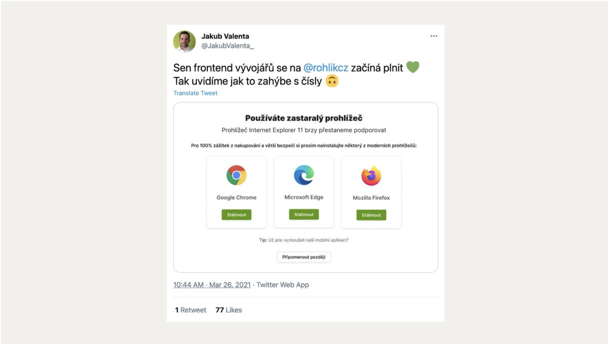
Je samozřejmě otázka, při jakém podílu na návštěvnosti je možné podporu prohlížeče vzdát. Obvykle se má za to, že podpora prohlížečů s podílem pod 1 % se nevyplácí, ale záleží na více faktorech:

- Kolik času a peněz musíte do podpory investovat. Troufám si říct, že toto číslo bude u Exploreru vysoké, protože už na příkladu CSS gridu je vidět, jak moc odlišné řešení je pro Explorer nutné dělat.
- Jak důležití jsou uživatelé prohlížeče pro výdělečnost vašeho projektu.

Ten druhý bod je přitom klíčový. Vyfiltrujte si v Google Analytics tržby uživatelů Exploreru a porovnejte je se svými náklady.

Takto to pojali například vývojářky a vývojáři v Rohlík.cz. Podíl MSIE je na Rohlíku nízký, jenže přepočteno na tržby se to pořád vyplatí.

Pro každého frontendistu je ovšem podpora takto zastaralého prohlížeče otrava. V Rohlíku tedy uživatelům Exploreru ukazují hlášku motivující je ke změně prohlížeče.

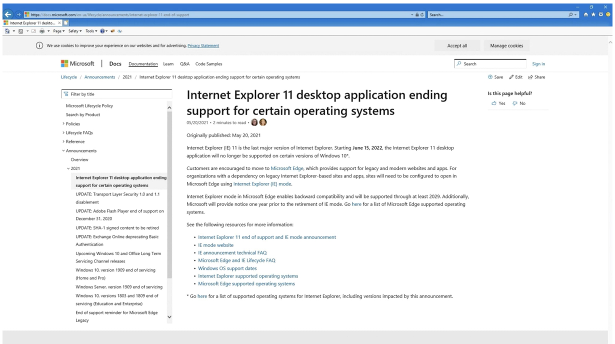


Máme vás rádi, ale nechtěli byste změnit prohlížeč? Hodně by nám to pomohlo. Zdroj: Twitter.

Ve schopnost těchto hlášek snížit podíl zastoupení starých prohlížečů zase tak moc nevěřím, ale jako součást širšího strategického balíčku „přemlouvání uživatelů" smysl mají.

Exploreru končí podpora od Microsoftu

Oficiální ukončení podpory mají v Microsoftu naplánované na polovinu června 2022.



Oznámení o ukončení podpory MSIE zobrazené v MSIE. Hurá, stránka se nerozpadla!

Termín je pro uživatele Exploreru od Microsoftu nastavený velkoryse, ale předpokládám, že uživatelé ve spolupráci s webaři ukončí podporu Internet Exploreru daleko dříve.

Odpočívej v pokoji. A děkujeme za vše dobré, milý Internet Explorere

Explorer byl dobrý prohlížeč – svého času. V době, kdy přišla verze 4, udělal Microsoft prohlížečovou revoluci a otočil chod dějin webu ve svůj prospěch.

Nicholas C. Zakas připomíná ve svém výborném článku „The innovations of Internet Explorer“ dobré věci, které nám MSIE přinesl:

> **Věřte tomu nebo ne, ale Internet Explorer 4–6 je do značné míry zodpovědný za webovou vývojařinu, jak ji známe dnes. Řada proprietárních funkcí se stala de facto standardy a poté oficiálními standardy, přičemž některé z nich skončily ve specifikaci HTML5.**

Z jeho textu, zmiňujícího mnoho inovací Exploreru, jsem vybral pár bodů, které stojí za připomenutí:

1. Zlepšil práci s DOMem tím, že umožnil programový přístup ke každému prvku na stránce prostřednictvím `document.all`, což byl předchůdce `document.getElementById()`.
2. Ve verzi 3 přidal Microsoft kromě tehdy populárních rámů (`<frame>`) také novou vlastní značku: `<iframe>` pro vnitřní rámy, dodnes velmi populární, například pro vkládání komponent třetích stran, jako je reklama nebo přehrávače videí z YouTube.
3. Internet Explorer 3 byl také první prohlížeč, který vsadil na CSS. V té době totiž konkurenční společnost Netscape prosazovala alternativní návrh, JavaScript Style Sheets (JSSS).
4. Často se zapomíná, že v Microsoftu nepřímo vymysleli Ajax. Zpracování XML na straně klienta bylo součástí implementace XMLHttpRequest, která byla poprvé představena jako součást rozšíření ActiveX v páté verzi Internet Exploreru.

Celý text Nicholase C. Zakase najdete na odkaze: vrdl.in/msieinn

Nezmínil jsem zde rok 2011, kdy Microsoft implementoval CSS grid. Ale to už znáte z první kapitoly.

Od doby příchodu Firefoxu a pak nástupu Chromu nebo Edge od Microsoftu jsme se s Explorerem my webaři už ale jen trápili, takže odchod oslavíme.

Důstojně, s respektem k nebožtíkovi, ale oslavíme.

Podpora layoutů v prohlížečích

To, že se v e-booku o layoutech v CSS zabýváme starým Explorerem, samozřejmě není jen tak. V této podkapitole se chceme zaměřit na problémy prohlížečů s podporou flexboxu, gridu a spol.

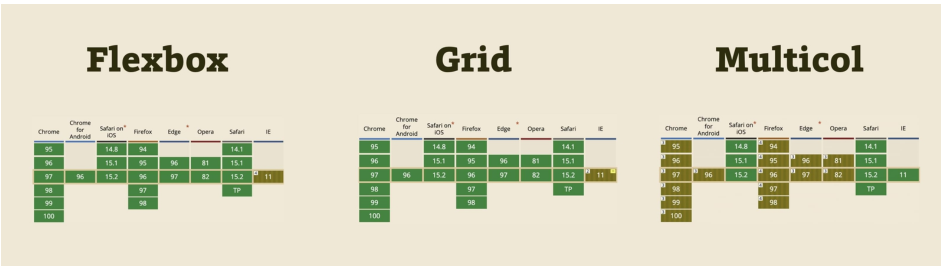
Pokud Explorer na svých projektech nepodporujete, pak i tuto podkapitolu směle přeskočte. Explorer vám velmi zkomplikuje využití CSS gridu, všechny ostatní systémy layoutu tam ale jsou použitelné.

Ty drobné chyby, které se vztahují na moderní prohlížeče a mřížku, flexbox, vícesloupcový layout a zarovnání boxů, spíše nestojí za řeč a budu je dále rozebírat v kapitolách s příručkou k jednotlivým vlastnostem.

„Nové systémy layoutu v CSS jsou tak úžasné, že to jistě musí mít nějaký háček," mohl by si někdo myslet.

Autorovi téhle myšlenky bychom museli přiznat jistou dávku zdravého skepticismu. Nebo dokonce nemalou životní zkušenost s vývojem webů.

Je to tak, milé čtenářky a milí čtenáři, určitá „ale" zde jsou. Musíme ovšem s úlevou prohlásit, že oproti dřívějšímu stavu je přítomnost chyb spojených s flexboxem nebo gridem v prohlížečích minimální.



Čím více zelené, tím více podpory. Jiný odstín (v barvě zelenohnědé) značí, že prohlížeč má nějaký problém. Ale ne s vámi, neberte si to osobně. Zdroj: [CanIUse.com](https://caniuse.com).

Co konkrétně znamenají ty zelenohnědé obdélníky u IE v případě gridu a flexboxu a skoro všude v případě multicol na obrázku?

- Jediným větším a hůře odstranitelným problémem je slabá podpora CSS gridu ze strany Internet Exploreru 11, i to se ale dá částečně řešit Autoprefixerem.
- U flexboxu máme prakticky plnou podporu, jen v MSIE 11 si musíme dát pozor na pár chyb.
- Podpora vícesloupcového layoutu je naopak v Exploreru výborná. Moderní prohlížeče si ale většinou hůř rozumějí s vlastnostmi `break-*`, určenými pro ovládání zalamování vnitřních prvků do sloupců.

Mimochodem, poprvé jsem tady zmínil skvělý nástroj Autoprefixer, který automaticky dodává CSS prefixy i další kód pro starší prohlížeče. Předpokládám jeho obecnou znalost, ale pokud jste se s ním zatím nesetkali, přidal jsem o něm podkapitolu do poslední, „přílohové" kapitoly.

V následujícím textu vezmeme nové systémy layoutu jeden po druhém a k jejich podpoře v prohlížečích si něco povíme.

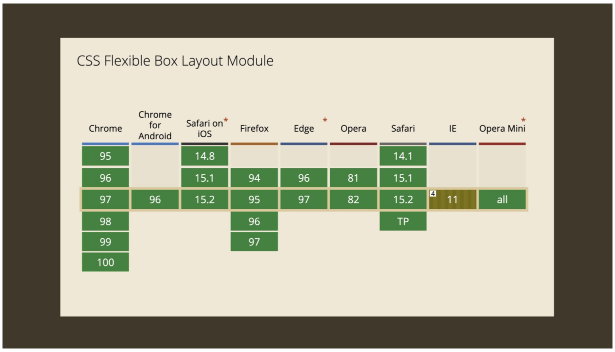
Flexbox a „flexboty" v MSIE 11

Když jsem začal flexbox před lety používat, bylo to trochu jako procházka minovým polem.

Za každým řádkem kódu mohla číhat nekompatibilita v některém prohlížeči. Člověk potřeboval detektor min a tím se stala stránka „Flexbugs" od Philipa Waltona. Jde o seznam 17 chyb v prohlížečích, které tehdy potřeboval znát každý, kdo pokouší osud psaním flexboxového kódu. vrdl.in/fbugs

Znalost některých chyb byla tak zásadní, že jsem je svého času musel učit na svých školeních. A při té příležitosti jsem jim začal říkal česky: *flexboty*. Každý děláme boty. Flexboty jsou chyby, které zanechali výrobci prohlížečů při implementaci flexboxu.

Když jsem během psaní tohoto textu srovnával současný stav s tím dřívějším, došel jsem k radostnému poznání.



Zelená se tráva… Podpora flexboxu v moderních prohlížečích je takřka bezchybná. Zdroj: CanIUse.com.

Je to tak, jak vidíte na obrázku. Takřka všechny chyby v moderních prohlížečích jsou opravené. Zůstaly jen ty navázané na Internet Explorer 11.

Podpora flexboxu je prakticky plná. U chyb, které v následujícím textu uvádím, se jedná o boty menší velikosti. Ale když už si o podpoře flexboxu povídáme takto detailně, je potřeba se o nich zmínit.

Flexboty v moderních prohlížečích

Moderní prohlížeče, tedy všechny kromě Internet Exploreru, byly zodpovědné za pět chyb ze seznamu Philipa Waltona.

To už ale dávno neplatí, dle mých testů zůstaly jen dvě málo důležité chyby.

- *Některé elementy nemohou být flex kontejnerem (flexbug #9).*
  Dříve to platilo i pro `<fieldset>` a `<button>` ve všech prohlížečích, což je naštěstí opravené. Zůstává jen málo nepříjemné omezení použití flexboxu na prvku `<summary>` v prohlížeči Safari. V tomto případě stačí použít vložený `<div>` jako kontejner pro rozvržení flexboxem.
- *Zalamované elementy na inline flexboxu přetékají z rodiče (flexbug #14).*
  Jde o kombinaci použití `flex-flow:column wrap` a `display:inline-flex`, takže poměrně vzácný scénář. Vnitřní prvky pak ve všech prohlížečích „vylezou“ z velikosti rodiče, i když by neměly. Je možné to obejít například pomocí nastavení `flex-direction:row` a změnou směru vykreslení zápisem vlastnosti `writing-mode`.

Pokud byste netušili, co je „inline flexbox“, jde o flexbox tvořený nastavením `display:inline-flex`. Uvnitř je možné dělat rozvržení, zvenčí jde o součást řádku textu. Více se tomu věnuji ještě v příručce vlastnosti `display` v deváté kapitole.

Můžete se podívat na řešení druhé uvedené flexboty.

CodePen: vrdl.in/r63gj

A co náš dědeček mezi prohlížeči?

Flexboty v MSIE 11

Tohle je zajímavější. Internet Explorer byl sice první prohlížeč, který moderní layouty naimplementoval, ale stejně jako všechny ostatní „prvoimplementace" šlo o pokus plný chyb.

Problémem MSIE tedy není množství chyb (chyby dělají všichni programátoři prohlížečů), ale způsob aktualizace.

Kdysi tak populární prohlížeč od Microsoftu vycházel v nových verzích z dnešního pohledu velmi pomalu, po letech, nikoliv měsících. A navíc – jedenáctá verze Exploreru je poslední a nikdo ji už aktualizovat nebude.

Pojďme na ty chyby, ať máme tu nepříjemnost za sebou.

Často tady budu mluvit o vlastnosti `flex-basis`. Pokud ji neznáte, vězte, že jde o podobnou věc, jako je vlastnost `flex`, tedy určení rozměrů položky. Zkratka `flex` toho umí jen o trochu víc.

1. Vlastnost `flex-basis` nezohledňuje `box-sizing:border-box` *(flexbug #7)*.
2. Vlastnost `flex` s nulovou `flex-basis` neplatí *(flexbug #4)*.
3. Vlastnost `flex-basis` neumí funkci `calc()` *(flexbug #8)*.
4. Položky flexboxu nemohou být `display:inline` *(flexbug #12)*.
5. Položky flexboxu se špatně zarovnají, když se užívá `max-width` *(flexbug #17)*.
6. Položky flexboxu lezou z kontejneru, který má `align-items:center` *(flexbug #2)*.
7. Vlastnost `min-height` na flex-kontejneru nefunguje *(flexbug #3)*.
8. Položky flexboxu nedodržují poměr stran *(flexbug #5)*.

9. Položky flexboxu neumí zarovnání pomocí `margin:auto` na příčné ose *(flexbug #15)*.

Máte přečteno? A máte z toho depresi? Chvilku počkejte.

Tyhle chyby detailně znát nemusíte. Většinu vaší práce s flexboxem neohrozí. Stačí jen vědět, že si v případě podivného chování MSIE 11 u flexboxu musíte vzpomenout na existenci stránky Flexbugs nebo tohoto textu. A pak ještě jednu věc.

Zkuste vynechat flex-basis

Když jsem se vývojářů na Twitteru ptal na jejich vychytávky spojené s flexboxem a gridem, Daniel Střelec napsal jednu, se kterou se ztotožňuji:

> **U flexboxu jsem se naučil definovat vždy kompletní zápis, tedy „flex: 1 1 auto“ (nespoléhat na default), a pokud to jde, tak používat vlastnost width místo flex-basis nebo obojí.**

Vysvětlím to. Zápis `flex:1 1 auto` je zkratka, která v prvním čísle definuje rozsah zvětšování položky, v druhém rozsah zmenšování a ve třetím výchozí velikost, která se ve flexboxu nastavuje vlastností `flex-basis`. Je možné ji nezapsat a ponechat výchozí hodnotu (`flex:auto`), což vám ale v případě nutnosti podpory MSIE nedoporučuji.

Dalším nutným vstřícným krokem je potřeba vyhnout se vlastnosti `flex-basis`. Často tedy stačí namísto ní použít `width` nebo `height` a všechno to dobře funguje. I v Exploreru.

Pokud máte tu smůlu, že s layouty začínáte a ještě pořád držíte podporu Exploreru, neděste se toho. Nic komplikovaného na tom není a používat flexbox i s podporou MSIE 11 je úplně v pohodě.

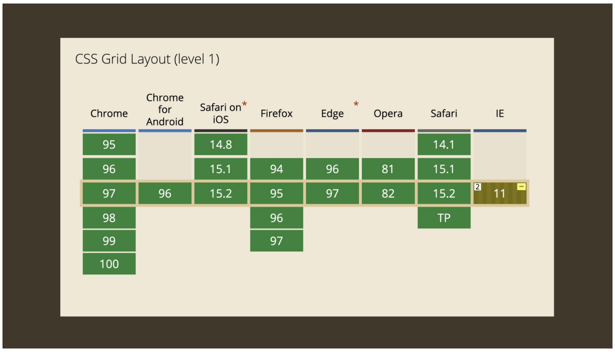
A co další chyby?

Pokud jste počítali, do celkových 17 chyb stále tři chybí. Ano, máte pravdu a vyhráváte… pobyt v Muzeu historie webových prohlížečů.

Zbývající tři boty má na svědomí Internet Explorer 10 a ten už dávno vyhynul. Více informací o podpoře je na CanIUse. caniuse.com/flexbox

Grid

V případě gridu se – daleko silněji než u flexboxu – musíme rozdělit na dvě skupiny vývojářek a vývojářů: Na ty, kteří nemusí Internet Explorer 11 podporovat. A pak na ty, kteří mají smůlu.



Podpora gridu v prohlížečích. Jiný odstín u MSIE nevěstí značí špatnou podporu. Zdroj: CanIUse.com.

Na obrázku vidíte podporu gridu v prohlížečích, které mají v ČR nad 0,5 % podílu trhu. Prohlížeče jsou seřazené podle používanosti.

Když jsem si na Twitteru dělal průzkum mezi vývojáři, vyšlo mi, že významná většina dává přednost flexboxu před gridem. Obávám se, že za to může komplikace jménem podpora gridu v MSIE.

Je potřeba říct, že i tenhle prohlížeč grid podporuje – a nepodporuje toho z něj vůbec málo: implicitní mřížku, funkci `repeat()`, funkci `minmax()` nebo klíčová slova `min-content` a `max-content`.

Na druhou stranu – jde jen o menší podmnožinu současné šíře vlastností toho, čemu říkáme CSS Grid Layout, navíc často jinak implementovanou.

Máme zde sice Autoprefixer, který „současný grid" umí překládat do podoby „IE gridu", ale jen částečně a navíc to vyžaduje další znalosti a schopnost tento nástroj bezchybně nastavit.

S gridem je to prostě v MSIE složité a já se vůbec nedivím lidem, kteří říkají: „Skoro na všechno mi stačí flexbox", i když pak flexbox používají pro situace, kde by byl výhodnější grid.

Jo, to když Explorer podporovat nemusíte, to je jiná písnička…

Gridbugs, boty v mřížce

Podobně jako první implementace flexboxu, také první enginy napsané pro vykreslování gridu byly v prohlížečích plné chyb.

A tak se známá propagátorka moderních rozvržení v CSS Rachel Andrew nechala inspirovat stránkou Flexbugs a vytvořila svého času její obdobu pro mřížku. Vznikly gridbugs. github.com/rachelandrew/gridbugs

Asi jste si všimli, že o té stránce píšu v minulém čase. Ke dni psaní zde vidím poslední změnu ze září 2017. Podobně jako u flexboxu, také u gridu postupně prohlížeče chyby odstranily.

Během přípravy pro psaní tohoto textu jsem poctivě prošel všech 14 chyb a podle všeho zůstává aktivní jen jedna bota, *gridbug #3* – chybějící podpora fragmentace.

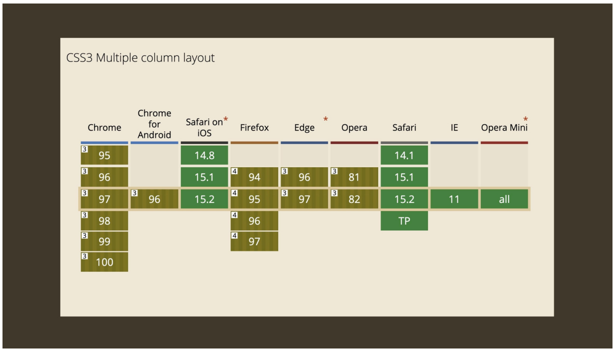
Jde o to, že prohlížeče v layoutu dělaném gridem špatně implementují vlastnosti jako `break-*`, kterými můžeme vynucovat konec stránky například v tiskové verzi.

Všechny ostatní chyby jsou, zdá se, opravené. Takže, když nebereme v úvahu Internet Explorer (jak úlevné!), CanIUse nás zaplaví zelenou barvou jako louka na jaře. caniuse.com/css-grid

Vícesloupcové rozvržení, CSS Multiple Columns

Multicolumn layout v CSS, takže sada specifikovaná kolem vlastnosti `column`, je na tom s podporou v prohlížečích poměrně dobře.

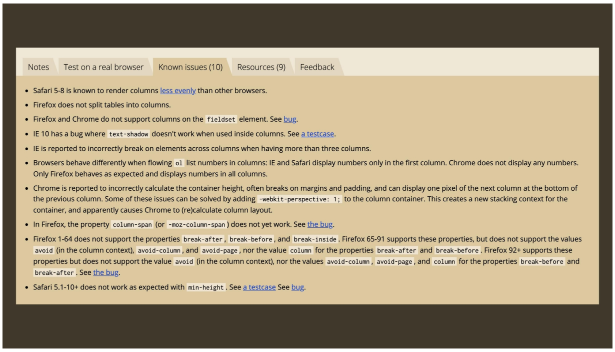
Zajímavé je, že tuto specifikaci trápí spíše nedodělky v moderních prohlížečích. Implementace v Exploreru je vlastně výborná.



Podpora vícesloupcového layoutu v prohlížečích. Zelená je na místech, kde byste to možná nečekali. Zdroj: CanIUse.com.

Chyby v Chromu a Firefoxu zde zůstávají už léta hlavně proto, že sloupcový systém rozvržení v CSS není právě často používaný, a tudíž není ani tak velký tlak na programátory prohlížečů.

Jsou to chyby spíše menšího kalibru, ale je jich poměrně dost.



Pytel blech. Drobné problémy ve vícesloupcovém layoutu. Zdroj: CanIUse.com.

Asi nejznámějším bugem, čili chybou v podpoře, je špatná podpora zalamování pomocí vlastností typu `break-*` v prohlížeči Chrome a všech, které z něj vycházejí. V kontextu tématu této knížky jde spíše o drobnost, ale je zde asi největší rozdíl mezi specifikací na papíře a reálným chováním prohlížečů.

Pokud bychom se dnes mohli bavit o nějakém „zabugovaném" systému layoutu, nebyl by to grid ani flexbox. Pokud jde o moderní prohlížeče, Černého Petra si bohužel vytáhlo vícesloupcové rozvržení. Je to samozřejmě proto, že CSS Multicol je výrazně nejméně používaným systémem rozvržení.

Více informací o něm najdete na našem oblíbeném webu CanIUse. caniuse.com/multicolumn

A co zarovnávání, CSS Box Align?

Čtvrtou specifikací z party systémů rozvržení v CSS je zarovnání boxů, které je nedílnou součástí navrhování layoutů v gridu a flexboxu.

Tenhle systém layoutu bohužel samostatný záznam na CanIUse nemá. Je to proto, že původně šlo o součást specifikace flexboxu, kde také na webu sledujícím podporu v prohlížečích vlastnosti CSS Box Alignment zůstaly.

Najdete je zde jako jednotlivé vlastnosti, když budete hledat text `align-` nebo `justify-`.

Tabulky podpory CSS Box Align v Internet Exploreru

Podpora v moderních prohlížečích je v zarovnávání vynikající, ale je potřeba zmínit, že i v tomto případě je Internet Explorer (MSIE) problematický. Opět hlavně v kombinaci s gridem.

Nepodporované vlastnosti jsou ty, které začínají slovem `place-`, dále vadí nepodpora `align-items` a `align-content`.

	Hlavní osa `justify-*`	**Příčná osa** `align-*`	**Oba směry** `place-*`
Zarovnání položek `*-items`	`justify-items` MSIE: ~~flex~~, ~~grid~~	`align-items` MSIE: flex, ~~grid~~	`place-items` MSIE: ~~flex~~, ~~grid~~
Zarovnání sebe sama `*-self`	`justify-self` MSIE: ~~flex~~, grid	`align-self` MSIE: flex, grid	`place-self` MSIE: ~~flex~~, ~~grid~~
Distribuce obsahu `*-content`	`justify-content` MSIE: flex, ~~grid~~	`align-content` MSIE: flex, ~~grid~~	`place-content` MSIE: ~~flex~~, ~~grid~~

Chybějící podpora `justify-items` i `justify-self` v MSIE u flexboxu je vlastnost, nikoliv bug. Tyto vlastnosti s flexboxem bohužel nelze kombinovat v žádném prohlížeči. Vysvětlím to v příručce k vlastnosti `justify-self`.

V případě, že podporujete MSIE, to je u gridu celkově složitější, viz následující podkapitola.

Explorer také nepodporuje novější hodnoty některých vlastností: `baseline` a `stretch`, vlastností `align-self` a `justify-content` nebo také `space-evenly` u `justify-content`.

O zarovnání boxů a jeho konkrétních vlastnostech se dočtete v sedmé kapitole.

Na layout typu masonry zatím čekáme

`align-tracks`, `justify-tracks` a další části layoutu typu masonry v CSS zatím žádný prohlížeč nepodporuje.

Layouty typu masonry v CSS tedy sledujte, ale pro praktické nasazení využijte jinou cestu.

Shrnutí podpory

Jak sami vidíte, o různých problémech systémů rozvržení v CSS v různých prohlížečích se dá popsat hodně papíru. A to jsem leccos vynechal.

Nicméně, důležitý je celkový dojem. Pokusím se to shrnout takto:

- Flexbox je víceméně bezproblémový. Při použití v Exploreru raději nahraďte `flex-basis` za `width` nebo `height`.
- Grid je v Exploreru problémový, musíte mít zvláštní znalosti. V moderních prohlížečích ale funguje skoro úplně bez potíží.
- Vícesloupcový layout umí zkomplikovat život menšími chybami, kterých je celkem dost.
- Zarovnání boxů je v případě flexboxu skoro bez zádrhelů všude, v případě gridu je to s MSIE opět složitější. Ve vícesloupcovém layoutu nejde vlastnosti Box Align použít.

Přístupnost: pozor na pořadí

Přístupnost je důležitá disciplína, s jejíž pomocí mohou vývojáři vyjít vstříc různým skupinám lidí a jež se jen zdánlivě týká pouze handicapovaných minorit, jako jsou zrakově postižení.

Díky přístupnosti se zkrátka mohou různé skupiny lidí dostat k informacím na webu bez velkých překážek.

V nových CSS layoutech může přístupnost pokazit poněkud kontroverzní možnost změny pořadí prvků ve stránce.

Přístupnost a pořadí ve flexboxu nebo gridu

Musím vás upozornit na to, že jakmile odlišíte pořadí zobrazování od pořadí v kódu, může se stát, že při ovládání z klávesnice (tabulátorem) nebo při použití nejrůznějších asistivních technologií (například odečítačů obrazovky) přestane uživatelům pořadí dávat smysl.

Také proto je ve specifikaci obsaženo toto důrazné varování:

> **Autoři musí použít změnu pořadí pouze pro vizuální, nikoliv logické přeskupování obsahu.**

Logické pořadí je zpravidla pořadí zápisu kódu a jeho využití si můžete představit v těchto kontextech konzumace stránky:

- *Roboti*
  Například stroje vyhledávačů, jako je Google. Roboti postupují podle pořadí v HTML nebo DOMu.
- *Sekvenční navigace stránkou*
  Tento typ procházení využívají například uživatelé odečítačů obrazovky nebo uživatelé, kteří z nějakého důvodu nemohou použít jiný způsob navigace – ať už z důvodu trvalého či dočasného postižení rukou, jako je třeba zlomenina.
- *Hlas a jiná média*
  Přeskupení vizuálního pořadí nezmění řazení v nevizuálních médiích, například v řeči.

Může se tedy stát, že někdo, kdo se naviguje pomocí klávesnice, bude procházet odkazy na vašem webu a najednou odskočí z dolní do horní části dokumentu, protože tam je další položka logického pořadí.

Ve specifikaci mřížky se dále píše:

> **CSS grid dává autorům velkou moc přeskupit dokument. Nejedná se však o náhradu za správné uspořádání zdroje dokumentu.**

Něco vám řeknu. Specifikace má pravdu. Pokud chcete pro přístupnost něco udělat, rozhodně dbejte na to, aby pořadí v kódu dávalo smysl v případě, kdy byste stránku četli bez stylů.

Dotčené vlastnosti

Problém se týká všech CSS vlastností, které mohou v nových systémech rozvržení ovlivnit vizuální pořadí:

- Vlastnost `order`, která změní způsob automatického umísťování položek.
- Deklarace `grid-auto-flow:dense`, jež automaticky přeskupí položky jinak, než jsou uvedeny v DOMu.
- Vlastnost `grid-area`, která umístí položky do konkrétního místa mřížky a opět nemusí respektovat pořadí ve zdroji.
- Vlastnost `flex-direction` a hodnoty, které převracejí pořadí – `row-reverse` a `column-reverse`.

Možností, jak přeskupit obsah, je samozřejmě více a vztáhnout to můžeme i na starý dobrý `float`, takže tento text berte jako obecné varování.

Příklad

Pojďme si to ukázat na jednoduchém demu postaveném na vlastnosti `order` a flexboxu se čtyřmi položkami:

```
<div class="container">
  <a class="item item--1" href="#">Item 1</a>
  <a class="item item--2" href="#">Item 2</a>
  <a class="item item--3" href="#">Item 3</a>
  <a class="item item--4" href="#">Item 4</a>
</div>
```

Jak vidíte, tentokrát jsem vyměnil `div`y za odkazy, a to proto, abychom mohli obsahem navigovat pomocí tabulátoru.

Kontejner (`.container`) je obyčejný flexbox, ale za ukázání kódu stojí předpis pro třetí položku:

```
.item--3 {
  order: -1;
}
```

Původní pořadí (1, 2, 3, 4) se tedy při prohlížení stránky v prohlížeči změní na 3, 1, 2, 4.

Jenže navigační pořadí je prohlížečem stále bráno podle HTML.

Zkuste si to naživo v CodePenu: vrdl.in/ykj23.

Zachránce tabindex? Leda kulový

Někteří z vás si určitě řekli, že situaci může přeci zachránit vlastnost `tabindex`, atribut HTML, který nastaví pořadí navigace pomocí tabulátorů:

```
<div class="container">
  <a class="item item--1" href="#" tabindex="2">Item 1</a>
  <a class="item item--2" href="#" tabindex="3">Item 2</a>
  <a class="item item--3" href="#" tabindex="1">Item 3</a>
  <a class="item item--4" href="#" tabindex="4">Item 4</a>
</div>
```

V prohlížeči to po přidání atributů `tabindex` může vypadat nadějně.

Pořadí navigace je nyní správné, protože odpovídá logickému uspořádání položek na obrazovce:

Typ řazení	**Pořadí**
HTML, DOM	1, 2, 3, 4
Vizuální	3, 1, 2, 4
Navigační, logické	1, 2, 3, 4

Zdá se, že problém jsme vyřešili. Pořadí v HTML a pořadí navigační se neliší.

Jenže chyba lávky! Otázka totiž zní: Jak moc je `tabindex` pro tyto případy v praxi použitelný?

Atribut `tabindex` totiž nastavuje pořadí pro *celý* dokument, takže jakmile bychom takto zapsali samostatnou komponentu a tu pak vkládali na různá místa DOMu, napácháme s `tabindex` více škody než užitku.

A pak – neumím si představit, že bychom kvůli jedné komponentě s `order` natvrdo měnili pořadí `tabindex` pro celou stránku.

Z mého pohledu je tedy `tabindex` pro opravu tohoto problému použitelný jen velmi omezeně.

Ale zkusit si to v CodePenu klidně můžete.

CodePen: vrdl.in/hz86d

Pomůže aria-flowto?

Léonie Watson v článku „Flexbox & the keyboard navigation disconnect" už před lety upozorňovala na vlastnost `aria-flowto`, která v rámci specifikace WAI-ARIA umožňuje právě lokální změnu navigačního pořadí. Je to prostě takový `tabindex` pro komponenty.

Hned jsem to vyzkoušel, ale zdá se, že stále platí to, co píše Léonie v článku z roku 2016: Tahle vlastnost má extrémně špatnou podporu v prohlížečích. Alespoň něco se v prohlížečích nemění…

CodePen: vrdl.in/dmou2

Pomůže nám vůbec něco? Ani ne

Odlišení pořadí navigačního od vizuálního je možné ve flexboxu a teď i gridu mnoha různými způsoby, tedy nejen vlastností `order`.

Bohužel se to zdá jako aktuálně nevyřešitelný problém, protože jej myslím nijak konkrétně neřeší specifikace, natož pak prohlížeče.

Odkážu vás na další zdroje, ale nic veselého se tam nedozvíte:

- Varování ve specifikaci flexboxu. vrdl.in/9e6zu
- Totéž ve specifikaci CSS gridu. vrdl.in/il128
- Adrian Roselli: „Source Order Matters“. vrdl.in/h0lgw
- Manuel Matuzovic: „The Dark Side of the grid (Part 2)“. vrdl.in/sju8i
- Rachel Andrew: „Content re-ordering and accessibility“. vrdl.in/1qad6
- Léonie Watson: „Flexbox & the keyboard navigation“. vrdl.in/42mb5

Komunitu, tedy vývojáře, lidi kolem webových specifikací a prohlížečů, zde ještě čeká dost práce. Jednoho by to v roce 2022 překvapilo.

Ponaučení do praxe zní: Jakmile použijete některou z vlastností, která rozpojuje pořadí vizuální od pořadí logického, přemýšlejte, jak velký vliv to bude mít na přístupnost a hlavně na vaše uživatele tam venku.

Rychlost: pozor na složitá rozvržení

V diskuzích pokročilejších vývojářek a vývojářů se občas objevuje téma „performance“ CSS gridu a flexboxu. Mají tím na mysli rychlost vykreslování.

Když byly tyhle systémy pro rozvržení stránek a komponent nové, poměrně dost se to v komunitě řešilo. Ale je to téma ještě dneska? Neřekl bych.

V textu dále se dozvíte, že flexbox je v extrémně složitých layoutech rychlejší. A je zde jedna věc, na kterou byste si měli dát pozor – na použití flexboxu pro rozvržení celé stránky nebo obzvlášť dlouhého obsahu, jako jsou komplexní články.

Je rychlejší grid, nebo flexbox? A není to jedno?

Před lety vznikaly studie, které se různými propracovanými metodami snažily zjistit, který systém rozvržení je pro vykreslování v prohlížeči lépe optimalizovaný.

Vyhrál to grid, teda flexbox, teda grid, teda flexbox.

Ano, takhle zmateně by se to dalo shrnout. V určitých typech layoutu exceluje flexbox, v jiných zase grid.

Podstatné ale je, že vy a vaši uživatelé to nejspíš nepoznáte. Studie byly skoro vždy ryze teoretické a postavené například na testech vykreslování desetitisíců prvků do stránky.

Za všechny uvedu jednu z Itálie. Vývojář Sandro Baccega testoval vykreslení 10 000 prvků do stránky. Pokud byste někdy něco takového potřebovali, vězte, že rychlejší to bude udělat flexboxem. V jeho testech to prohlížeči pomocí flexboxu zabralo něco kolem 40 ms, zatímco s gridem kolem 200 ms. Autor k tomu píše:

> **Pravděpodobně to vychází ze skutečnosti, že ve flexboxu nemohou prvky následujících řádků zasahovat do vykreslení prvků předchozích řádků, zatímco v CSS gridu tomu tak není.**

V druhém testu, vykreslování složitého celostránkového layoutu, dopadl flexbox i grid velmi podobně.

Takže pokud se nechystáte vykreslovat desítky tisíc prvků do layoutu e-shopu nebo prezentačního webu, toto pro vás užitečná informace nebude.

V případě zájmu běžte přímo na text „CSS grid vs Flexbox: Performance Evaluation“ na smc.it. vrdl.in/gridflexper

Ten složitý celostránkový layout italského testu nám ale může posloužit – když už ne jinak – jako pěkný oslí můstek k zajímavějšímu testu.

Pokud nemusíte, nepoužívejte flexbox pro celostránkové layouty

Jake Archibald, jeden z mých oblíbených odborníků na výkon frontendových aplikací, už v roce 2014 varoval před používáním flexboxu pro celostránkové layouty.

Ptáte se proč? Jeden důvod je výkon, ale daleko důležitější jsou neměnné principy, jak flexbox rozvržení počítá.

Mrkněme se spolu na demo, které k tomu připravil. Zde je zjednodušené HTML:

```
<div class="container">
  <div class="site-header">
    <div class="site-title">This is my site</div>
  </div>
  <div class="content">
    <div class="main">
      <h1>This is my article</h1>
    </div>
    <nav id="nav">
      <ul>
        <li><a href="#">Home</a></li>
      </ul>
    </nav>
    <aside>
      <p>Hi, I'm Clarence Quince…</p>
    </aside>
  </div>
</div>
```

Ano, jde o klasické třísloupcové rozvržení s obsahem uprostřed. Verze pro flexbox má toto CSS:

```
.container {
  display: flex;
  flex-flow: row;
}

nav {
  flex: 1;
  min-width: 118px;
  max-width: 160px;
}

.main {
  order: 1;
  flex: 1;
  min-width: 510px;
}

aside {
  flex: 1;
  order: 2;
  min-width: 150px;
  max-width: 210px;
}
```

Verze pro grid vypadá následovně:

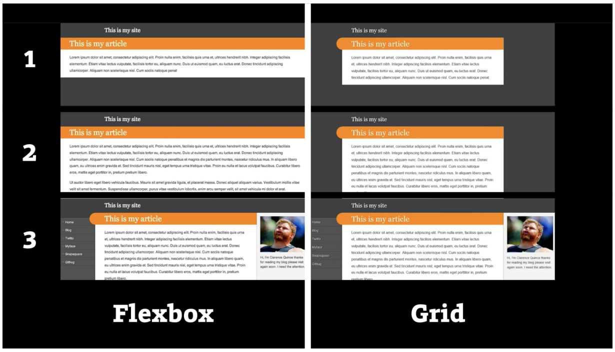
```
.container {
  display: grid;
  grid-template-columns: (nav) minmax(118px, 160px), (main) minmax(612px, 1fr),
    (aside) minmax(182px, 242px);
}

nav {
  grid-area: nav;
}

.main {
  grid-area: main;
}

aside {
  grid-area: aside;
}
```

A teď k těm rozdílům.



Rozdíly ve vykreslování gridu a flexboxu, na které poukazuje Jake Archibald. Ne, to není ten nafukující se pán na obrázku.

Ještě to popíšu, vypadá to asi následovně:

- V případě gridu je layout připravený už při první verzi vykreslené stránky a dále se již nemění. Kontejner určuje layout obsahu.
- Flexbox vykreslí nejprve hlavní část s textem. Pak si prohlížeč jakoby vzpomene, že zde jsou ještě dvě boční části, a vykreslí je. Dojde tedy k překreslení. Obsah určuje layout kontejneru.

Připomínám, že nechtěné překreslení layoutu uživatele rozhodně nepotěší a webu zhorší metriku Cumulative Layout Shift, kterou se zabývá Google v rámci svých Web Vitals. vrdl.in/cls

Jake Archibald ovšem varuje: „Nenechte se kvůli tomuto příspěvku flexboxem vyděsit." Má pravdu. Jeho test je postavený i na svou dobu na silně zpomalené rychlosti stahování a vykreslování. Dnes už rychlost mobilního připojení 2G nemá smysl testovat, používá se většinou pomalejší 4G. Jeho dema mi v roce 2021 už při sebevětším zpomalení testovacího zařízení a připojení k internetu žádné postřehnutelné nekalosti neukázala.

Nicméně, je nezpochybnitelný fakt, že flexbox i grid jsou vykreslovány zcela jiným způsobem:

- Flexbox se vykresluje od obsahu až k layoutu. Prohlížeč nejprve musí znát obsah, spočítat jeho rozměry a pak teprve vykreslit layout.
- Grid renderují prohlížeče tak, jak byste očekávali – nejprve layout, pak teprve obsah. Layout je proto vždy pevný jako beton.

Zcela konkrétně to popsal Bohumil Jahoda, autor skvělého JeČas.cz, kterého zde prostě a jednoduše ocituji:

> **Flexbox trpí tím, že musí čekat na stažení celého HTML nebo poskakuje. (…) Týká se to použití u dlouhého obsahu. Problém je v tom, že poslední element může totálně ovlivnit, jak se daná část vykreslí. Takže prohlížeč buď musí čekat na stažení celého potřebného HTML, nebo vykreslit neúplný obsah, což může vést k tomu poskakování.**

Příspěvek jsem přejal z Twitteru, jak jinak, kde si můžete přečíst celou diskuzi. vrdl.in/flexposk

Nejde ani tak o flexbox, jako o přednost obsahu při skládání layoutu

V článku najdete i demo situace, kdy se může stejně špatně jako flexbox pro komplexní layout vykreslovat i grid. Stačí, když při tvorbě rozvržení dáte přednost obsahu:

```
.container {
  display: grid;
  grid-template-columns:
    (foo) max-content,
    (bar) min-content,
    (hello) auto;
}

aside {
  grid-column: 4;
}
```

Stručně to popišme:

- Klíčová slova `max-content` a `min-content` dávají instrukci, aby se prvek nezvětšoval nad maximální nebo nezmenšoval pod minimální velikost obsahu.
- Klíčové slovo `auto` zde odpovídá `minmax(min-content, max-content)`, takže nejmenší i největší možné rozměry této buňky layoutu jsou opět na prohlížeči.
- Třída `container` určuje třísloupcový layout. Prvek `aside` v něm není definován. Přidáme jej dynamicky, což má za následek nechtěné překreslení layoutu.

O funkci `minmax()` a klíčových slovech `max-content` a `min-content` se více dozvíte v kapitole o gridu.

Vraťme se k varování Jakea Archibalda, ale nenechme si tím flexbox zprotivit. Problémy se pravděpodobně projeví jen na velmi komplexních stránkách a ještě na pomalých zařízeních.

Vychází z toho ale jedno ponaučení – pokud kódujete celostránkový layout ve flexboxu, nezapomeňte ho otestovat na nejhorším možném zařízení, které si ve vaší cílové skupině umíte představit.

Nebo pro tyhle účely – komplexní nebo celostránková rozvržení – prostě používejte grid.

Ještě vám dlužím odkaz na Archibaldův článek „Don’t use flexbox for overall page layout“. vrdl.in/6pajl

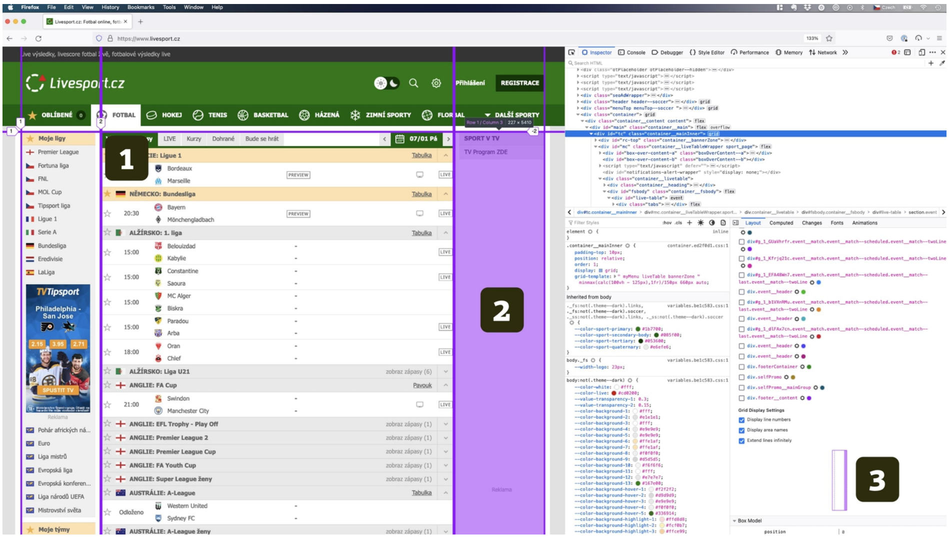
DevTools a CSS layout

Vsadím se, že už jen při listování touto knížkou jste si všimli, že rozvržení v CSS jsou sakra složitá věc.

Máme ale štěstí, že autoři nástrojů pro vývojáře to vidí také a v „devtools“ všech dnešních prohlížečů je řada nadstaveb, které nám práci s gridem a flexboxem usnadní.

Firefox, pak Chrome a nakonec i Safari

Lídrem trhu bychom v této oblasti mohli s klidným srdcem jmenovat autory Firefoxu.

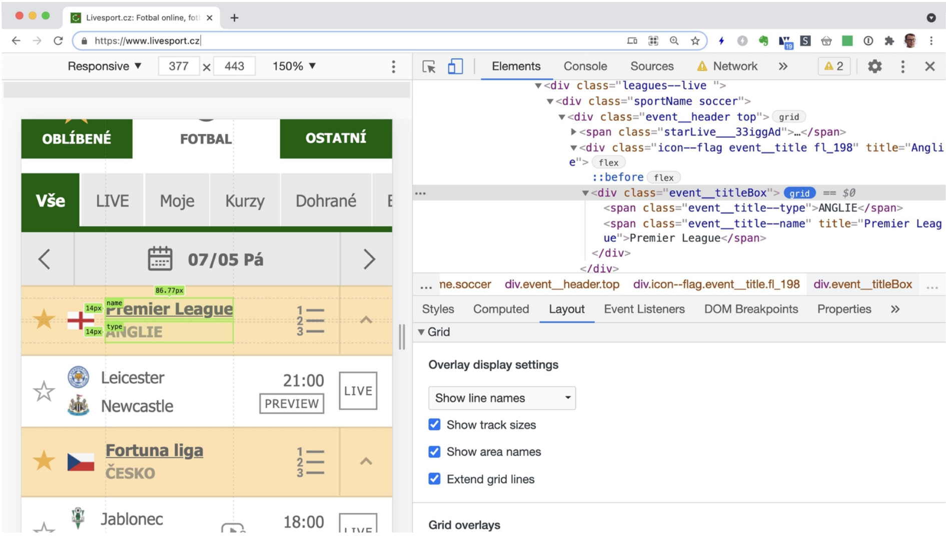


It works! CSS grid Inspector v DevTools Firefoxu používaný na Livesport.cz.

Už v březnu 2017, zároveň s implementací CSS gridu, do Firefoxu přibyl propracovaný CSS grid Inspector. Jediný prohlížeč nezávislý na velké korporaci v té době patřil v oblasti vývojářského pohodlí mezi lídry trhu.

Firefox ale od té doby bohužel prošel krizí, kdy musel snižovat počet lidí, kteří vývojářské nástroje navrhují a programují.

Někde jsem pod dojmy z nevyváženosti množství vývojářů pracujících na Firefoxu a Chromu napsal, že prohlížeč od Googlu všechny dobré vlastnosti Firefoxu rychle zkopíruje.



Jako dvojče. CSS grid na Livesport.cz pitvaný pomocí vývojářských nástrojů Chromu.

V tomto případě to ale trvalo mnoho let, až do podzimu 2020. Tehdy přišli autoři Chromu s vlastní verzí, pojmenovanou „CSS grid debugging tools“, velmi silně inspirovanou Firefoxem. vrdl.in/griddevchr

A co Safari? Jako skoro vždy – dlouho nic a pak to přišlo. V březnu 2021 představili velice podobný nástroj a nyní je možné grid zkoumat už i v Safari.

Co přesně můžete v prohlížečích zkoumat?

Pojďme se na tyhle nástroje podívat z větší blízkosti.

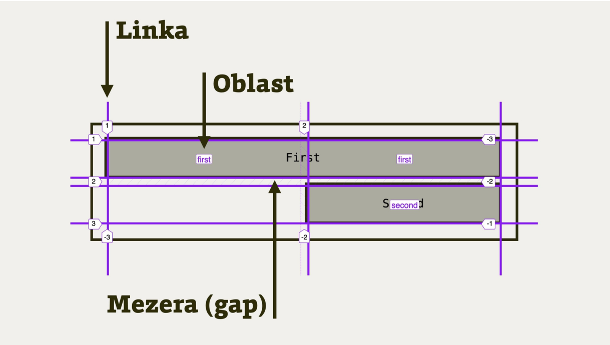
V Chromu zapnete zkoumání gridu nebo flexboxu takto:

1. Pusťte na stránce DevTools: F12 nebo Ctrl/Cmd+Alt/Option+I.
2. Otevřete záložku Elements. Prvky, které je možné takto ladit, mají štítek „grid“ nebo „flex“. Klikněte na štítek.

3. Na stránce se vám pak objeví překryvná vrstva, která nese více informací o daném rozvržení.
4. Ve vývojářských nástrojích navíc přibyla záložka „Layout“, kde je seznam všech flexových a gridových prvků, plus také další informace.

Pokud používáte Firefox, v návodu namísto záložky Elements dosadíte Inspector.

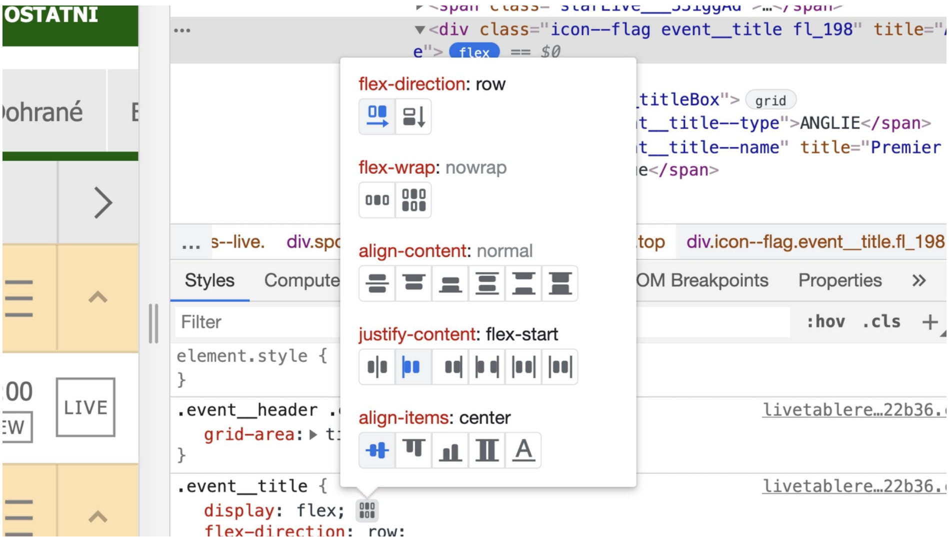
Osobně se mi zdá, že Firefox má v této oblasti pořád alespoň malý náskok. Zejména jejich překryvná vrstva pro ladění gridu je navržená velmi dobře.



Překryvná vrstva ve Firefoxu. Zkoumám tam tento CodePen: vrdl.in/othcx

V překryvné vrstvě gridu a flexboxu ve Firefoxu krásně vidím linky, oblasti a další pojmy, které jsou při návrhu gridu podstatné.

Chrome ale přišel s jednou novou vlastností, která se mi moc líbí.



Skoro jako WYSIWYG… Vizualizace a možnost přenastavení vlastností layoutu v Chromu.

Ve velmi složité oblasti, kterou představují vlastnosti gridu, flexboxu a zarovnávání, je tohle dobrý směr. Bez vizuálního ztvárnění začíná být velmi těžké si hodnoty vlastností zapamatovat.

Těším se, co nám konkurence prohlížečů ještě v tomhle směru přinese, a jsem zvědav, zda se Firefoxu podaří těmito chytrými vlastnostmi alespoň v některých věcech porážet rozjetý tým Chromu.

Zde jsou odkazy na texty představující tyto nástroje v jednotlivých prohlížečích:

- „Firefox 52 for developers“. vrdl.in/griddevfir
- „Chrome 87: What’s New In DevTools“. vrdl.in/griddevchr
- „WebKit: Introducing CSS grid Inspector“. vrdl.in/griddevsaf

Shrnutí osmé kapitoly

1. **Jaký podíl na českém trhu má MSIE začátkem roku 2022?**
   a) kolem 42 %
   b) kolem 10 %
   c) kolem 1 %
2. **Co je logické pořadí prvků?**
   a) vizuální pořadí, jak prvky vidíme ve stránce
   b) pořadí prvků v HTML
   c) pořadí prvků, jak se jimi prochází, např. pomocí klávesy Tab
3. **Jaké vlastnosti mohou měnit logické pořadí ve stránce?**
   a) například `order`, `grid-auto-flow`, `grid-area`
   b) například `flex`, `grid-basis`, `grid-shrink`
   c) například `flex`, `grid-grow`, `grid-shrink`
4. **Jakou vlastnost raději vynechat, pokud se chceme vyhnout potížím v Internet Exploreru?**
   a) `flex-basis`
   b) `display`
   c) `flex-shrink`
5. **Proč může komplikovanější flexboxový layout „probliknout“?**
   a) layout může ovlivnit každá položka, počítá se totiž z obsahu
   b) layout odhaduje prohlížeč podle denní doby
   c) jde o chyták, tento problém vzniká jen u rozvržení gridem

---

Řešení: 1. c), 2. c), 3 a), 4 a), 5. a).

Martin Michálek
CSS: MODERNÍ LAYOUT

Redakce a jazyková korektura: Petr Jediný Novotný, pjnovotny.cz
Obrázky, schémata a vnitřní sazba: Martin Michálek
Obálka a spolupráce na grafice knihy: Petr Šťastný, raist.cz
Spolupráce na webu: Jan Kočiš, SUPERKODERS
Foto autora: Oldřich Hrb, oldrichhrb.cz

V knize jsou použita písma Capita a Foro od Dietera Hofrichtera (hoftype.com) a InconsolataGo od Rapha Leviena (levien.com).

Tisk: PBtisk a.s.

Vydal Martin Michálek – Vzhůru dolů

Verze 1.0, březen 2022

ISBN:
Tištěná kniha: 978-80-88253-07-5
PDF: 978-80-88253-08-2
MOBI: 978-80-88253-09-9
ePUB: 978-80-88253-10-5